TENTAMEN

Programmering Grundkurs (HI1900)

Skrivtid 8:15-13:15

Måndagen 18 oktober 2010

Tentamen består av 8 sidor

Hjälpmedel
Förutom dator med installerad *Code::Blocks*,
Utforskaren, *Acrobat reader* och *Notepad* (inga andra program),
den kurslitteratur som använts under kursen, samt egna anteckningar,
programlistningar och böcker. Dock inga egna disketter, CD-ROM eller USB-minne.

- Under `W:\PROV\C` finns program- och datafiler som kan komma till användning vid lösandet av uppgifterna. Kopiera över dessa till ditt konto.
- Till alla uppgifter ska ett program levereras i form av källkod (C eller CPP-fil). Dina bidrag lägger du i en katalog i roten på `H:`. Katalogen ska ha samma namn som prefixet i din *mailadress*. Exempelvis för Kalle Kula: `HDI02KEKA`. Namnen på lösningarna ska ges `UPPG1.C` till `UPPG8.C`. De är endast dessa filer som kommer att bedömas.

Rättningen görs genom att programmen körs ett antal gånger för olika indata. Om resultatet överensstämmer med det förväntade bedöms programmet som korrekt och ger 2 poäng. Om ett program ej kan kompileras utan fel, är det knappast troligt att det kommer att ge några poäng. I det fall där programmet läser från och eller skriver till en fil, testas programmet oftast med en annan fil än den bifogade.

Betygsgränser (HI1900):

Poäng	14-16	12-13	10-11	8-9	7	6
Betyg	A	B	C	D	E	Fx

Resultatet anslås på kursens hemsida `ingforum.haninge.kth.se/c` i kodat skick.

Lycka till!

Johnny Panrike, Håkan Strömberg

Uppgift 1. Krysset

Skriv ett program som, efter att ha frågat efter storleken, skriver ut en kvadrat på skärmen, fylld med minustecken (-), utom i diagonalerna som ska markeras med plustecken (+). Storleken hos kvadratens sida ska kunna variera i intervallet [1...24] Ett körningsexempel:

```
Storlek ? 10
+--------+
-+------+-
--+----+--
---+--+---
----++----
----++----
---+--+---
--+----+--
-+------+-
+--------+
```

Uppgift 2. Ord med bokstäver i bokstavsordning.

På filen finns de ord i SAOL (Svenska akademiens ordlista) som inte innehåller vare sig **Å**, **Ä** eller **Ö** (för enkelhetens skull). Orden har återgivits med versaler (stora bokstäver).

Skriv ett program som tar reda på hur många ord det finns på filen `SAOL.txt`, där bokstäverna befinner sig i bokstavsordning. BELOPP, är exempel på ett sådant ord. Däremot inte BELYSA.

Filen inleds med ett tal `n`, som anger hur många ord den innehåller. Därefter följer `n` ord med ett ord på varje. Inget ord innehåller fler än 30 bokstäver.

Körningsexempel

```
Det finns 281 sådana ord
```

Uppgift 3. Vokaler, behövs de?

Frågan är: kan man gissa sig till ordet även då vokalerna tagits bort? Givet följande huvudfunktion:

```
int main(void){
  char ord[31];
  int antalvokaler;
  laesIn(ord);
  taBortVokaler(ord,&antalvokaler);
  skrivUtAsterisker(ord);
  printf("%d borttagna vokaler \n",antalvokaler);
}
```

- Programmet ska ta emot en sträng innehållande upp till 30 gemena (små) bokstäver i intervallet `a...z` genom funktionen `laesIn`.
- Programmet ska sedan från denna sträng avlägsna samtliga vokaler, (`a`, `e`, `i`, `o`, `u`, `y`), och samtidigt returnera detta antal, genom funktionen `taBortVokaler`.
- Programmet ska sedan skriva ut den återstående strängen, helt omgiven av asterisker (`*`), genom funktionen `skrivUtAsterisker`, se körninsexemplet
- Programmet avslutas med att skriva ut antalet avlägsnade vokaler.

Din uppgift är nu att skriva de tre funktionerna, `laesIn`, `taBortVokaler` och `skrivUtAsterisker`, så att programmet producerar ett resultat likt det i körningsexemplet. Observera att du inte får ändra på någonting i `main`! Bygg vidare på koden given i `uppg3.c`.

```
Strängen ? programmeringskurs
***************
*prgrmmrngskrs*
***************
5 borttagna vokaler
```

Uppgift 4. Riksdagsvalet 2010

På den binära filen `riksval2010.dat` finns uppgifter om resultatet från riksdagsvalet 2010 uppdelat på Sveriges 290 kommuner. Följande postbeskrivning, som behandlar en kommun, har använts:

```
struct kommuntyp{
  char namn[24];
  int laen;
  int roster[11];
};
```

`namn` innehåller kommunens namn. `laen` innehåller att tal som motsvarar tillhörande län, se figur 1. `roster` innehåller antalet röster på olika partier samt antalet ogiltiga röster, se figur 2.

1	Blekinge län	2	Dalarnas län	3	Gotlands län
4	Gävleborgs län	5	Hallands län	6	Jämtlands län
7	Jönköpings län	8	Kalmar län	9	Kronobergs län
10	Norbottens län	11	Skåne län	12	Stockholms län
13	Södermanlands län	14	Uppsala län	15	Värmlands län
16	Västerbottens län	17	Västernorrlands län	18	Västermanlands län
19	Västra Götalands län	20	Örebro län	21	Östergötlands län

Figur 1: *Tabell över Sveriges län och tillhörande nummer*

0	Moderata Samlingspartiet	1	Centerpartiet
2	Folkpartiet liberalerna	3	Kristdemokraterna
4	Arbetarepartiet-Socialdemokraterna	5	Vänsterpartiet
6	Miljöpartiet de gröna	7	Sverigedemokraterna
8	Övriga partier	9	Ogiltiga röster - blanka
10	Ogiltiga röster - övriga		

Figur 2: *Innehållet i* `roster` *med tillhörande index*

Skriv ett program som frågar efter numret för ett län och som tar reda på och skriver namnet på den kommun i givet län som har den lägsta procentuella andelen röster på *Sverigedemokaterna*. Observera att de ogiltiga rösterna (9 och 10) inte räknas in i denna andel. Ett körningsexempel:

```
Vilket län ? 11
Lund har lägsta andelen SD, 4.98 %, i län 11
```

Uppgift 5. Tentamen i C-programmering

En praktisk tentamen i C har 8 uppgifter. De svårare uppgifterna brukar komma sist. Studenterna som följer en kurs i programmering förväntas lägga ner cirka 20 arbetstimmar i veckan på att träna C-programmering under 7 veckor och kanske 10 timmar i tentamensveckan. Totalt cirka 150 timmar.

Låt oss anta att man kan mäta *förmågan att programmera* i procent (ett tal mellan 0 och 100) och att 150 timmars träning ger 100% förmåga. Vi antar här att ökningen är linjär, alltså 1.5 timmars träning ger 1% programmeringsförmåga.

Låt oss nu kalla den uppnådda förmågan till C-programmering för p (som då är ett tal mellan 0.0 och 100.0). Vi antar vidare att sannolikheten att man klarar den första uppgiften är $0.9 \cdot p$, sannolikheten att man klarar den andra är $0.85 \cdot p$, sannolikheten att man klarar den tredje är $0.8 \cdot p$, och så vidare med 0.05 mindre chans för varje uppgift. Sannolikheten att man klarar den sista uppgiften blir således $0.55 \cdot p$.

För att klara en tentamen ska man klara minst 4 av uppgifterna. För att få genomföra en komplettering för godkänt ska man klara 3 av uppgifterna. Skriv ett program som med hjälp av 100 000 simuleringar, beräknar sannolikheten för att man klarar tentamen samt sannolikheten att man får komplettera, för ett givet antal arbetstimmar. Körningsexempel:

```
Antal timmar tränat ? 150
Sannolikhet att klara tentamen är 96.54 procent.
Sannolikhet att få komplettering är 2.94 procent.
```

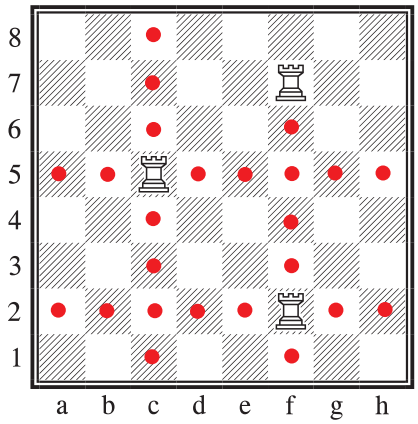
Uppgift 6. Bankkonton

På filen `konton.txt` finns uppgifter om ett antal bankkonton och transaktioner dem mellan. Filen inleds med ett tal $n < 100$ som anger antalet konton som ska hanteras. Därefter följer n rader med två tal på varje, *kontonummer* (ett fyr- eller femsiffrigt tal) och *kontoställning* i kronor. Direkt efter dessa rader följer ett tal t som anger antalet *transaktioner* följt av t rader med tre tal på varje, *från vilket konto pengarna kommer*, *till vilket konto pengarna ska*, samt *beloppet*. Alla belopp är heltal.

Skriv ett program som bestämmer kontoställningarna för de nk kontona efter att transaktionerna genomförts. Ett körningsexempel:

```
Konto  Belopp
-----  ------
14705  124053
25704   24468
 2715    2995
11841   76184
```

Uppgift 7. Schacktornen



Figur 3:

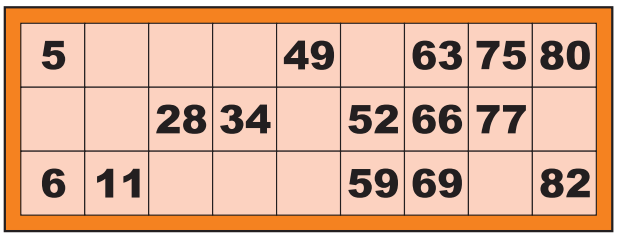
Schack spelas på ett bräde med 8×8 rutor. Man spelar genom att flytta pjäser på brädet. Olika pjäser kan flyttas på olika sätt. Tornet är kanske den lättaste pjäsen att flytta, den kan endast flyttas rakt i alla riktningar, hur långt som helst. Vi ser i figur 3, tre torn. För två av tornen har vi ritat ut röda prickar som visar de rutor som de kan nå genom att flytta dem. Det tredje tornet har vi inte ritat ut prickar för.

Man säger att två pjäser *garderar* varandra, om båda pjäserna kan flyttas till den ruta, som den andra pjäsen står på. I figur 3 ovan, med tre torn, garderar de två tornen till höger varandra, medan det tredje inte garderas av eller garderar något av de andra.

Beräkna sannolikheten för att tre slumpvis utplacerade torn alla är garderade av åtminstone ett annat torn. Simulera utplaceringen av tre torn på ett schackbräde 100 000 gånger. Programmet ska ange den beräknade sannolikheten med två decimaler. Körningsexempel:

```
Den beräknade sannolikheten är 9.68 %
```

Uppgift 8. Bingo



5				49		63	75	80
		28	34		52	66	77	
6	11				59	69		82

Figur 4: *En typisk bingobricka*

Bingo är som bekant ett slags lotteri eller turspel. Till den här varianten finns ett antal bingobrickor, liknande de i figur 4. Varje bricka består av tre rader med fem tal (`1...90`) på varje. Varje spelare har en egen bricka. I en urna finns 90 bollar, numrerade `1...90`. Spelledaren drar en kula i taget från urnan och meddelar numret. Spelaren markerar dragna nummer som förekommer på dennes bricka. Vinner gör den som först lyckas ”fylla” en hel rad. Spelaren har fått BINGO!

På filen `brickor.txt` finns beskrivning av $n < 100$ brickor. Filen inleds med ett tal `n` som anger hur många. Därefter följer $3n$ rader med 5 tal på varje. En bricka beskrivs alltså med tre på varandra följande rader. Brickorna tänks numrerade 1 till `n` i den ordning de återfinns på filen.

På filen `urna.txt` finns en rad med talen 1 till 90 i den ordning de kommer att dras av spelledaren.

Skriv ett program tar reda på vilken bricka som vinner, först lyckas få en rad med fem dragna nummer, och efter hur många dragna kulor detta inträffar. Om flera spelare samtidigt får BINGO räcker det med att ditt program presenterar en av dessa.

Ett körningsexempel:

```
Bricka nr 29 vann efter 14 dragna nummer
```

Uppgift 1. Krysset

```c
#include <stdio.h>
int main(void){
  int i,j,storlek;
  printf("Storlek ? ");
  scanf("%d",&storlek);
  for(i=1;i<=storlek;i++){
    for(j=1;j<=storlek;j++)
      if(i==j || i+j-1==storlek)
        printf("+");
      else
        printf("-");
    printf("\n");
  }
}
```

Uppgift 2. Ord med bokstäver i bokstavsordning.

```c
#include <stdio.h>
#include <string.h>

int koll(char s[]){
  int i;
  for (i=1;i<strlen(s);i++)
    if (s[i-1]>s[i])
      return 0;
  return 1;
}

int main(void){
  FILE *fil;
  int i,antal,funna=0;
  char ordet[50];

  fil=fopen("ord.txt","rt");
  fscanf(fil,"%d",&antal);
  for (i=1;i<=antal;i++){
    fscanf(fil,"%s",ordet);
    if (koll(ordet)){
      printf("%s\n",ordet);
      funna++;
    }
  }
  printf("Vi har funnit %d\n",funna);
  fclose(fil);
}
```

Uppgift 3. Vokaler behövs de?

```c
#include <stdio.h>
#include <string.h>
void laesIn(char s[]){
  printf("Ordet ? ");
  scanf("%s",s);
}

void taBortVokaler(char s[],int *b){
  char vokaler[]="aeiouy",t[31]="";
  int i,j,n=0,ok;
  for (i=0;i<strlen(s);i++){
    ok=0;
    for (j=0;j<6;j++)
      if (vokaler[j]==s[i])
        ok=1;
    if (!ok)
      t[n++]=s[i];
  }
  t[n]='\0';
  *b=strlen(s)-strlen(t);
  strcpy(s,t);
}

void skrivUtAsterisker(char s[]){
  int i;
  for (i=1;i<=strlen(s)+2;i++)
    printf("*");
  printf("\n*%s*\n",s);
  for (i=1;i<=strlen(s)+2;i++)
    printf("*");
  printf("\n");
}

int main(void){
  char ord[31];
  int antalbort;
  laesIn(ord);
  taBortVokaler(ord,&antalbort);
  skrivUtAsterisker(ord);
  printf("%d borttagna vokaler \n",antalbort);
}
```

Uppgift 4. Riksdagsvalet 2010

```c
#include <stdio.h>
#include <string.h>

struct kommuntyp{
  char namn[24];
  int laen;
  int roster[11];
};

int main(void){
  FILE *fil;
  int antal,i,j,sum,laen;
  double sd, sdmin=100.0;
  char sdminnamn[24];

  struct kommuntyp kommun;
  printf("Vilket laen ? ");
  scanf("%d",&laen);

  fil=fopen("riksval2010.dat","rb");
  fseek(fil,0,SEEK_END);
  antal=ftell(fil)/sizeof(struct kommuntyp);
  rewind(fil);

  for (i=0;i<antal;i++){
    fread(&kommun,sizeof(struct kommuntyp),1,fil);
    if (kommun.laen==laen){
      sum=0;
      for (j=0;j<9;j++)
        sum=sum+kommun.roster[j];
      sd=(double)kommun.roster[7]/sum*100;
      if (sd<sdmin){
        sdmin=sd;
        strcpy(sdminnamn,kommun.namn);
      }
    }
  }
  printf("%s har lägsta andeln SD, %.2f %%, i län %d\n",
         sdminnamn,sdmin,laen);
}
```

Uppgift 5. Tentamen i C-programmering

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(void){
  int tim,i,j,godk=0,kompl=0,klarat;
  double p1,p2,s;
  printf("Hur många timmar har du tränat ? ");
  scanf("%d",&tim);
  srand(time(0));
  p1=tim/150.0;
  for (i=1;i<=100000;i++){
    klarat=0;
    for (j=1;j<=8;j++){
      p2=p1*(0.95-j*0.05);
      s=(double)rand()/RAND_MAX;
      if (s<=p2)
        klarat++;
    }
    if (klarat>=4)
      godk++;
    if (klarat==3)
      kompl++;
  }
  printf("Sannolikhet för godkänd är %.2f\n",godk/1000.0);
  printf("Sannolikhet för komplettering är %.2f\n",kompl/1000.0);
}
```

Uppgift 6. Bankkonton

```c
#include <stdio.h>

void flytta(int k[][2],int f,int t,int b,int nk) {
  int i,fran,till;
  for(i=0; i<nk; i++) {
    if(f==k[i][0])
      fran=i;
    if(t==k[i][0])
      till=i;
  }
  k[fran][1]-=b;
  k[till][1]+=b;
}

int main(void) {
  FILE *fil;
  int konton[100][2],i,fran,till,belopp,nk,nt;

  fil=fopen("konton.txt","rt");
  fscanf(fil,"%d",&nk);
  for(i=0; i<nk; i++)
    fscanf(fil,"%d %d",&konton[i][0],&konton[i][1]);
  fscanf(fil,"%d",&nt);
  for(i=0; i<nt; i++) {
    fscanf(fil,"%d %d %d",&fran,&till,&belopp);
    flytta(konton,fran,till,belopp,nk);
  }
  for(i=0; i<nk; i++)
    printf("%7d %7d\n",konton[i][0],konton[i][1]);
  fclose(fil);
}
```

Uppgift 7. Schacktornen

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

void slumpa(int p[][2]){
  int i,j,klart;
  for(i=0;i<3;i++){
    do{
      p[i][0]=rand()%8+1;
      p[i][1]=rand()%8+1;
      klart=1;
      for(j=0;j<i;j++)
        if(p[i][0]==p[j][0] && p[i][1]==p[j][1])
          klart=0;
    }while(!klart);
  }
}

int garderar(int p[][2]){
  int i,j,k,antal=0;
  for(i=0;i<2;i++)
    for(j=i+1;j<3;j++)
      for(k=0;k<2;k++)
        if(p[i][k]==p[j][k])
          antal++;
  return antal;
}

int main(void){
  int plats[3][2],i,gard=0;
  srand(time(0));
  for(i=1;i<=100000;i++){
    slumpa(plats);
    if(garderar(plats)>=2)
      gard++;
  }
  printf("Den sökta sannolikheten %.2f\n",gard/1000.0);
}
```

Uppgift 8. Bingo

```c
#include <stdio.h>
#include <conio.h>

int main(void) {
  int brickor[100][3][6];
  FILE *bfil,*ufil;
  int b,r,k,klart=0,nr,antal=0,n;

  bfil=fopen("brickor.txt","rt");
  ufil=fopen("urna.txt","rt");
  fscanf(bfil,"%d",&n);

  for(b=0; b<n; b++)
    for(r=0; r<3; r++) {
      for(k=0; k<5; k++)
        fscanf(bfil,"%d",&brickor[b][r][k]);
      brickor[b][r][5]=0;
    }
  fclose(bfil);

  while(1) {
    fscanf(ufil,"%d",&nr);
    antal++;
    for(b=0; b<n; b++)
      for(r=0; r<3; r++)
        for(k=0; k<5; k++)
          if(brickor[b][r][k]==nr) {
            brickor[b][r][5]++;
            if(brickor[b][r][5]==5) {
              printf("Bricka nr %d vann efter %d dragna nummer\n",
                      b+1,antal);
              goto ut;
            }
          }
  }
ut:
  fclose(ufil);
}
```